International Journal of Scientific Research and Engineering Development-– Volume 9 Issue 4, July – August 2026
Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

Autonomous AI Agent for Business Intelligence: A Multi-Agent Orchestration Framework

Akula Kavya Sri*, Dr. P. Chiranjeevi**
*(Department of Computer Science and Engineering, Amrita Sai Institute of Science and Technology (Autonomous), Paritala, Andhra Pradesh, India
Email: kavyasriakula010@gmail.com)
**(Department of Computer Science and Engineering, Amrita Sai Institute of Science and Technology (Autonomous), Paritala, Andhra Pradesh, India
Email: csehod@amritasai.org.in)

----------**********----------

Abstract:

Traditional Business Intelligence (BI) platforms depend on human analysts to prepare datasets, detect anomalies, configure visualizations, and interpret results in business language. This manual dependency creates an insight-to-action gap that delays strategic decisions by days or weeks. This paper presents Agentic BI, an autonomous multi-agent framework that automates the complete analytics lifecycle through seven specialized cooperative agents orchestrated over a centralized state controller. Built on Python, FastAPI, and Google Gemini 1.5 Pro, the system performs schema detection, autonomous data cleaning, Z-score and IQR-based anomaly isolation, KPI computation, programmatic chart generation, predictive forecasting, and LLM-driven strategic narrative synthesis. Experimental evaluation on a 21 MB enterprise benchmark dataset comprising over 102,400 records demonstrates end-to-end processing in 9.8 seconds with 100 percent pass rate across thirteen test scenarios. Results indicate that agentic orchestration significantly reduces analytical latency while preserving data integrity, democratizing prescriptive business intelligence for non-technical stakeholders.

Keywords — Agentic AI, Business Intelligence, Multi-Agent Systems, Large Language Models, Autonomous Analytics, Data Pipeline Automation

----------**********----------

I. INTRODUCTION

Organizations today generate vast volumes of structured transactional data, yet converting these records into timely strategic decisions remains a labor-intensive process. Business Intelligence (BI) platforms have traditionally served as the interface between raw data stores and executive reporting, offering dashboards, key performance indicators (KPIs), and historical trend visualizations. However, conventional BI workflows depend heavily on human analysts to prepare datasets, configure transformations, detect anomalies, and interpret results in business language [1], [2].

Industry studies consistently report that data preparation consumes approximately 80 percent of an analyst's working time, leaving limited capacity for prescriptive interpretation [3]. Manual extraction, cleaning, and reporting introduce latency measured in days or weeks rather than minutes. Furthermore, descriptive dashboards answer what happened but rarely explain why a metric changed or what corrective action should follow. This insight-to-action gap is particularly acute for small and medium enterprises (SMEs) that lack dedicated data science teams [4].

Recent advances in Large Language Models (LLMs) and Multi-Agent Systems (MAS) offer a fundamentally different approach to analytics

 ©IJSRED: All Rights are Reserved

automation. LLMs such as Google Gemini can synthesize statistical summaries into narrative executive reports, while specialized autonomous agents can perform discrete pipeline tasks with deterministic accuracy [5], [6]. Rather than relying on a single monolithic model, a collaborative network of task-specific agents mirrors the division of labor found in professional analytics departments.

This paper presents Agentic BI, an autonomous multi-agent framework designed to automate the end-to-end business intelligence lifecycle. The system orchestrates seven specialized agents—covering ingestion, cleaning, anomaly detection, KPI identification, visual discovery, forecasting, and strategic reasoning—over a centralized state controller implemented with Python and FastAPI. Google Gemini 1.5 Pro serves as the reasoning engine that transforms validated statistical outputs into prescriptive markdown reports suitable for non-technical stakeholders.

The primary contributions of this work are: (1) a decoupled three-tier architecture that isolates heavy computation from the presentation layer; (2) a sequential agentic pipeline with shared state management ensuring data integrity across modules; (3) deterministic statistical algorithms combined with constrained LLM prompting to reduce hallucination risk; and (4) empirical validation demonstrating sub-10-second end-to-end processing for enterprise-scale datasets exceeding 21 MB.

The remainder of this paper is organized as follows. Section II reviews related work in traditional BI, self-service analytics, and agentic AI. Section III presents system analysis and requirements. Section IV describes the proposed architecture and agent framework. Section V details implementation and algorithms. Section VI reports experimental results and discussion. Section VII concludes the paper and outlines future research directions.

II. II. LITERATURE REVIEW

Business intelligence has evolved through three distinct generations since the emergence of data warehousing in the 1990s. First-generation BI centered on Online Analytical Processing (OLAP) and centralized reporting infrastructure. These systems required specialized analysts to author complex Extract, Transform, Load (ETL) pipelines and SQL queries, resulting in rigid reporting cycles tied to batch warehouse refreshes [7].

Second-generation self-service BI tools, exemplified by Tableau and Microsoft Power BI, democratized dashboard creation through drag-and-drop interfaces during the 2010s [1]. Non-technical users could explore data visually without writing code. Nevertheless, self-service platforms still place the burden of data cleaning, schema alignment, and interpretation on the end user. Visualization capabilities improved substantially, but the underlying preparation overhead remained largely unchanged.

Third-generation augmented analytics incorporated automated machine learning for forecasting and anomaly highlighting. Platforms began suggesting insights proactively based on statistical models. However, augmented tools typically lack contextual reasoning capabilities—they can flag an outlier but cannot articulate its business significance or recommend strategic responses in natural language [2].

A. A. Generative AI and LLMs in Analytics

The Transformer architecture introduced by Vaswani et al. [10] revolutionized natural language processing and enabled large-scale language models capable of few-shot reasoning [11]. In analytics contexts, LLMs function as narrative synthesis engines that translate numerical aggregates into executive prose. Arora [2] demonstrated that generative AI can transform structured BI outputs into stakeholder-ready reports, though unconstrained models remain susceptible to hallucination when operating on raw unverified data.

Google Gemini 1.5 extends multimodal reasoning across extended context windows, making it suitable for synthesizing compact statistical payloads rather than entire datasets [5]. This architectural constraint motivates local pre-aggregation before LLM invocation—a design principle adopted in the present framework.

B. B. Multi-Agent Systems in Data Science

 ©IJSRED: All Rights are Reserved

Multi-Agent Systems decompose complex workflows into cooperating specialized units. Chen and Lin [6] surveyed collaborative AI architectures for data science and concluded that task decomposition improves both accuracy and maintainability compared with monolithic models. Agents communicate through shared state objects, enabling sequential validation and parallel analysis where dependencies permit.

Ramirez [4] traced the evolution from conversational chatbots to autonomous business analysts, arguing that agentic pipelines represent the next operational paradigm for enterprise analytics. The Agentic BI framework presented herein applies this paradigm by simulating a virtual data science team through seven domain-specific agents governed by a central orchestrator.

C. C. Comparative Analysis

Table I summarizes operational characteristics across first-generation BI, self-service BI, and the proposed agentic platform. The comparison highlights reductions in user effort, automation of data cleaning, and transition from descriptive to prescriptive insight delivery.

TABLE I
COMPARATIVE ANALYSIS OF BI PARADIGMS

Feature	Trad. BI	Self-Serv. BI	Proposed
User effort	Very high	Medium	Low
Data cleaning	Manual ETL	Manual	Autonomous
Insight type	Descriptive	Visual	Prescriptive
Time to report	Days/weeks	Hours	Seconds
Reasoning	Human	Human	LLM agent

The literature review reveals a persistent integration gap: advanced visualization tools and powerful language models exist independently, yet few systems combine both within an autonomous end-to-end pipeline. Agentic BI addresses this gap by chaining deterministic statistical processing with constrained generative reasoning under human-in-the-loop governance.

III. III. SYSTEM ANALYSIS AND REQUIREMENTS

System analysis establishes the functional boundary, feasibility constraints, and quality attributes for the proposed platform. The analysis contrasts existing manual BI workflows with the autonomous agentic alternative and formalizes requirements through a Software Requirements Specification (SRS) aligned with stakeholder needs.

D. A. Existing System Limitations

In conventional BI environments, analysts manually extract datasets from operational systems, apply cleaning scripts or spreadsheet macros, generate charts, and compose interpretive summaries. This linear human chain introduces inconsistency because different analysts may apply divergent cleaning rules or subjective narrative framing. Manual processes are error-prone, resource-intensive, and poorly suited to recurring reporting cadences required by modern enterprises.

Self-service platforms reduce dependency on IT departments but transfer preparation complexity to business users who may lack statistical training. Neither approach provides autonomous anomaly remediation, programmatic chart selection, or machine-generated strategic recommendations without substantial manual configuration.

E. B. Proposed System Overview

The proposed Agentic BI system replaces the linear analyst workflow with a parallel collaborative agent network. The platform is self-healing—automatically correcting data type inconsistencies and imputing missing values—and self-explaining—generating narrative interpretations of statistical findings. Users upload structured CSV or XLSX files and receive a dashboard with charts and an executive markdown report without writing queries or transformation scripts.

F. C. Feasibility Study

Technical feasibility was established using mature open-source components: Python 3.10+ with Pandas and NumPy for data manipulation, FastAPI for asynchronous web services, Scikit-learn for forecasting, Matplotlib and Seaborn for

 ©IJSRED: All Rights are Reserved

visualization, and the Google Gemini API for language synthesis [8], [9]. Economic feasibility is supported by negligible API costs relative to analyst labor for routine reporting. Operational feasibility follows from an intuitive web interface requiring no programming knowledge. Table II summarizes the feasibility assessment.

TABLE II
FEASIBILITY STUDY SUMMARY

Type	Status	Justification
Technical	Feasible	Mature Python and FastAPI stack
Economic	Feasible	Low API cost vs. analyst labor
Operational	Feasible	Simple upload-and-report UI
Schedule	Feasible	Modular agent development

G. D. Functional Requirements

Functional requirements define mandatory system behaviors derived from stakeholder interviews and comparative analysis of incumbent BI tools. Table III enumerates core functional components and their specifications.

TABLE III
FUNCTIONAL REQUIREMENTS

ID	Requirement	Description
FR-1	File ingestion	CSV/XLSX upload with schema detection
FR-2	Data cleaning	Type-specific missing value imputation
FR-3	Anomaly detection	Z-score and IQR outlier flagging
FR-4	KPI computation	Variance-based driver scoring
FR-5	Visualization	Auto trend, bar, and heatmap charts
FR-6	Forecasting	Regression and time-series projection
FR-7	Narrative report	Gemini-based executive summary

H. E. Non-Functional Requirements

Performance requirements specify end-to-end processing of 20 MB datasets within 15 seconds under standard development hardware. Accuracy requirements mandate 100 percent preservation of non-erroneous records during cleaning. Security requirements dictate in-memory processing with session-scoped aggregate clearance after report delivery. Usability requirements emphasize minimalist dashboard design accessible to business managers without technical training [12].

I. F. Hardware and Software Requirements

TABLE IV
HARDWARE AND SOFTWARE REQUIREMENTS

Component	Specification
Processor	Intel Core i5 (8th Gen) or higher
Memory	8 GB min.; 16 GB recommended
Storage	256 GB SSD
Backend	Python 3.10+, FastAPI, Pandas
ML/AI	Scikit-learn, Google Gemini 1.5 Pro
Visualization	Matplotlib, Seaborn (Agg backend)

IV. IV. SYSTEM ARCHITECTURE AND AGENT FRAMEWORK

The Agentic BI platform adopts a three-tier architecture that decouples presentation, computation, and data handling. This separation prevents user interface blocking during intensive dataframe operations and enables independent scaling of backend agents.

J. A. Three-Tier Architecture

The presentation tier delivers a FastAPI-served web dashboard rendered through Jinja2 templates. Asynchronous JavaScript polling provides real-time progress feedback as each agent completes its sub-task, offering transparency into the autonomous pipeline. The logic tier hosts the Agent Orchestrator and seven specialized workers connected through a Shared State Controller implemented as a session-scoped dictionary tracking dataframe status, anomaly findings, KPI aggregates, and chart artifacts. The data tier employs in-memory Pandas dataframes rather than persistent relational storage to minimize I/O latency during vectorized transformations.

 ©IJSRED: All Rights are Reserved

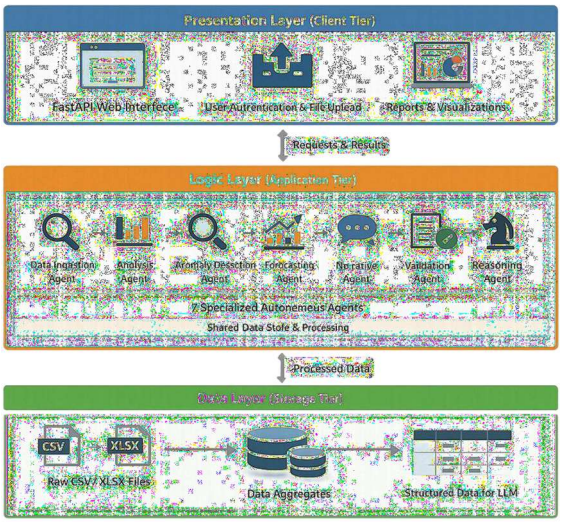

Fig. 1 Three-tier architecture of the Agentic BI platform

K. B. Specialized Autonomous Agents

Intelligence is distributed across seven autonomous modules executing in a validated sequential workflow with selective parallelization during statistical profiling.

The Data Ingestion Agent detects file encoding, parses multi-sheet workbooks, and maps initial column schemas. The Cleaning and Standardization Agent applies median imputation for numerical columns and mode or label substitution for categorical fields. The Statistical Anomaly Detection Agent implements Z-score and Interquartile Range (IQR) filters to isolate extreme values. The KPI Identification Agent computes variance-based business drivers across dimensional attributes.

The Visual Discovery Agent programmatically maps variables to chart types—line charts for temporal series, bar charts for categorical comparisons, and heatmaps for correlation matrices. The Predictive Forecasting Agent applies Scikit-learn linear regression and time-series models to project future performance. The Strategic Reasoning Agent constructs structured prompts from aggregated findings and invokes Google Gemini to generate executive markdown reports with overview, anomaly analysis, and actionable recommendations.

L. C. Use Case Model

Fig. 2 illustrates system boundaries and actor interactions. The primary actor is the business analyst or manager who uploads datasets and consumes finished reports. The internal orchestrator manages agent execution, while Google Gemini acts as an external reasoning service. The use case model emphasizes a hands-off workflow: minimal user input produces a complete analytical deliverable.

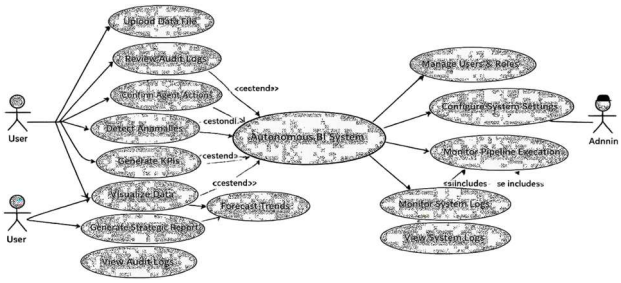

Fig. 2 Use case diagram of the Agentic BI system

M. D. Sequence and Workflow Design

Fig. 3 depicts the chronological agent handshake. After file upload, the orchestrator initializes shared session state and invokes the Cleaning Agent. Cleaned dataframes feed parallel Anomaly and KPI agents. Upon completion, aggregated payloads are formatted and transmitted to Gemini. The controller bundles returned narrative text with generated charts for dashboard rendering.

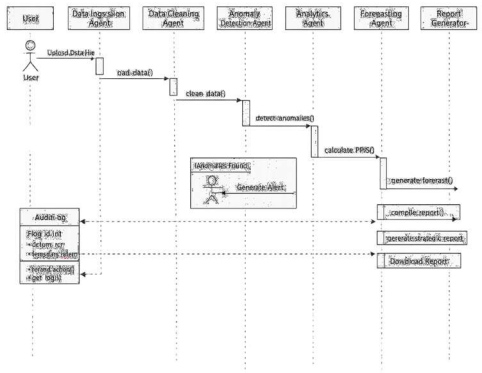

Fig. 3 Sequence diagram of the multi-agent BI pipeline

Fig. 4 presents the end-to-end workflow infographic summarizing ingestion, cleaning, anomaly auditing, KPI calculation, forecasting, visualization, and AI-driven report compilation stages.

 ©IJSRED: All Rights are Reserved

Fig. 4 End-to-end agentic BI workflow

N. E. Data Flow Design

Data Flow Diagrams (DFDs) formalize information movement across system processes. Level 0 treats the entire application as a single process exchanging raw datasets and analytical deliverables with the user. Level 1 decomposes processing into ingestion, agentic computation, insight synthesis, and visualization rendering. Level 2 details agent-level sub-processes operating on in-memory store D1.

Table V defines principal data flows. The Statistical_Payload JSON object carries KPI summaries and anomaly records from computational agents to the reasoning module, ensuring structured handoff without exposing raw row-level data to the LLM.

TABLE V
DATA FLOW DICTIONARY

Data Flow	Path	Description
Raw_Dataset	User → Ingestion	Uncleaned CSV/XLSX input
Cleaned_DataFrame	Cleaning → Analysis	Analysis-ready records
Statistical_Payload	Agents → Reasoning	KPI and anomaly JSON
Markdown_Report	Gemini → Renderer	Executive narrative text
Final_Dashboard	Renderer → User	Charts and report view

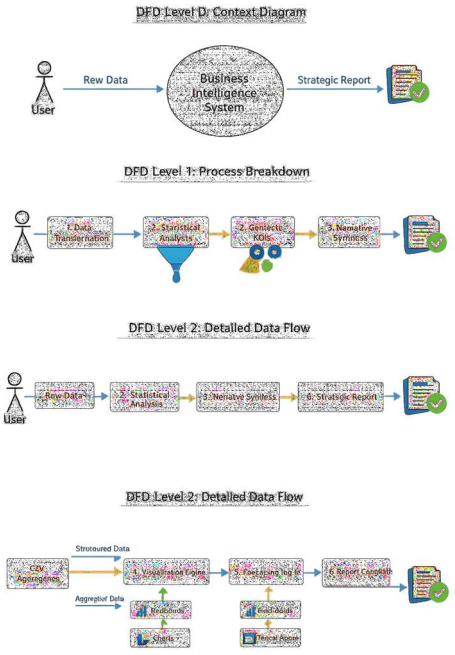

Fig. 5 Level-1 data flow diagram of the Agentic BI system

V. V. IMPLEMENTATION AND ALGORITHMS

Implementation follows a modular Python architecture emphasizing deterministic statistical processing prior to generative synthesis. This ordering constrains the LLM to interpret verified aggregates rather than infer patterns from raw records, mitigating hallucination risk.

O. A. Technology Stack

FastAPI provides asynchronous request handling and automatic OpenAPI documentation. Pandas and NumPy execute vectorized dataframe operations on datasets exceeding 100,000 rows. Scikit-learn supplies regression and forecasting primitives [9]. Matplotlib's Agg backend enables headless chart rendering with Base64 encoding for web embedding [13]. Virtual environment isolation manages dependency versioning across development and deployment targets.

P. B. Data Cleaning Algorithm

The Cleaning Agent scans all columns for null values and applies type-specific imputation. Numerical fields receive median imputation to remain robust against skewed distributions identified later by the Anomaly Agent. Categorical

 ©IJSRED: All Rights are Reserved

fields receive mode imputation or substitution with a uniform Unknown label to preserve row completeness without discarding valid records.

Q. C. Anomaly Detection Formulation

The Anomaly Agent applies the standard score (Z-score) for parametric outlier detection. For each numerical column, the agent computes the arithmetic mean μ and standard deviation σ, then evaluates Equation (1) for every observation x:

$$Z = (x - \mu) / \sigma$$

Observations satisfying $|Z| > 3$ are flagged at 99.7 percent confidence under normality assumptions and recorded in an Anomaly Report forwarded to the Reasoning Agent. Complementary IQR filtering handles non-Gaussian distributions by flagging values outside 1.5 times the interquartile range beyond the first and third quartiles.

R. D. Visualization Mapping Logic

The Visual Discovery Agent selects chart types based on column metadata rather than static templates. Temporal columns trigger line charts for trend analysis. Categorical versus numerical pairings produce bar or pie charts. Numerical correlation pairs generate heatmaps or scatter plots. This rule-based mapping ensures context-appropriate visuals without manual chart configuration [14].

S. E. Prompt Engineering Strategy

The Strategic Reasoning Agent employs few-shot constrained prompting to guide Gemini output. The system prompt instructs the model to analyze only provided KPI and anomaly aggregates, structure responses into Executive Summary, Key Performance Drivers, Anomaly Analysis, and Strategic Recommendations sections, and refrain from external knowledge injection. Token limit constraints are addressed by local group-by aggregation before API invocation, transmitting summarized insights rather than full datasets.

T. F. Orchestration Pipeline

The orchestrator executes an asynchronous pipeline: ingestion loads the source file; cleaning sanitizes records; anomaly and KPI agents profile the cleaned dataframe in parallel where dependencies permit; the forecasting agent projects trends; the reasoning agent synthesizes narrative output; and the renderer assembles the dashboard response. Each agent emits status signals upon completion, enabling frontend progress tracking and audit logging.

VI. VI. RESULTS AND DISCUSSION

System validation encompassed unit testing of individual agents, integration testing of inter-agent handshakes, and end-to-end system testing from file upload through report delivery. Performance benchmarking measured latency across small, medium, and large dataset configurations.

U. A. Testing Methodology

Unit tests verified mathematical correctness of imputation logic, Z-score computation, and aggregation functions. Integration tests confirmed JSON schema compatibility between KPI outputs and the Reasoning Agent prompt constructor. System tests executed the complete pipeline against a 21 MB benchmark dataset comprising over 102,400 transaction rows on an Intel Core i5 system with 8 GB RAM.

V. B. Test Case Results

Thirteen enterprise test scenarios were executed with 100 percent pass rate. Table VI summarizes representative test cases covering ingestion, cleaning, anomaly detection, visualization, LLM synthesis, error handling, and data consistency validation.

TABLE VI
REPRESENTATIVE TEST SCENARIOS AND RESULTS

ID	Scenario	Expected	Result
TC-01	21 MB CSV upload	< 5 s ingestion	Pass (2.1 s)
TC-02	Null imputation	All nulls filled	Pass
TC-03	Outlier detection	|Z| > 3 flagged	Pass
TC-04	Chart generation	3 chart types	Pass
TC-05	LLM synthesis	Markdown report	Pass
TC-06	Invalid format	Error returned	Pass
TC-07	Data consistency	Report matches data	Pass
TC-08	Concurrent	Stable	Pass

 ©IJSRED: All Rights are Reserved

	users	response	

W. C. Performance Benchmarking

Table VII presents end-to-end latency profiles across dataset sizes. Small datasets under 1 MB completed in 3.1 seconds. Medium datasets of 10 MB required 6.4 seconds. The 21 MB benchmark completed in 9.8 seconds, satisfying the sub-10-second performance objective. Ingestion latency scaled linearly with file size while agent processing dominated total time for larger volumes.

TABLE VII
PERFORMANCE BENCHMARK RESULTS

ID	Size	Rows	Total (s)
TC-BI-01	< 1 MB	4,500	3.1
TC-BI-02	10 MB	48,000	6.4
TC-BI-03	21 MB	102,400	9.8

X. D. User Interface and Generated Outputs

Fig. 6 shows the user dashboard featuring drag-and-drop upload and multi-stage progress tracking. Fig. 7 displays agent execution logs confirming sequential handshake validation. Figs. 8 and 9 present automatically generated visualizations selected by the Visual Discovery Agent. Fig. 10 illustrates the strategic narrative report including executive summary and SWOT analysis sections produced by the Reasoning Agent.

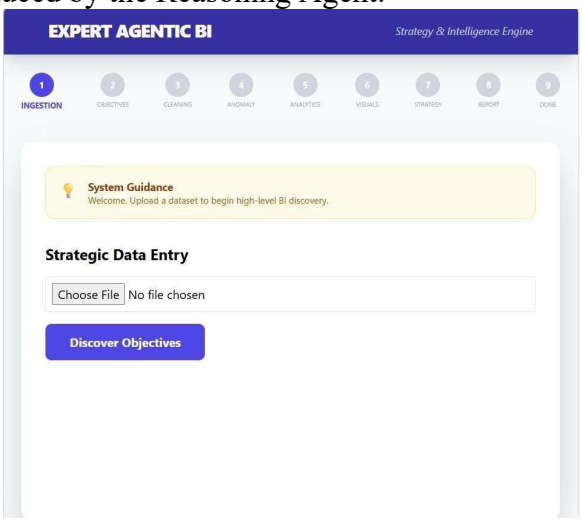

Fig. 6 User dashboard with file upload and progress tracking

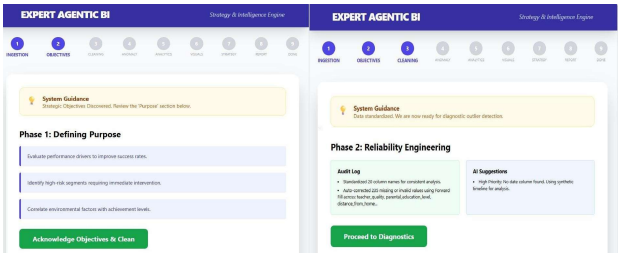

Fig. 7 Agent execution logs showing pipeline handshake status

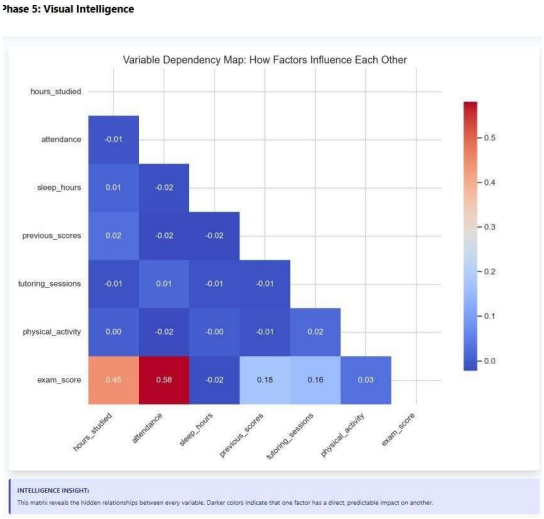

Fig. 8 Automated trend visualization generated by the Visual Discovery Agent

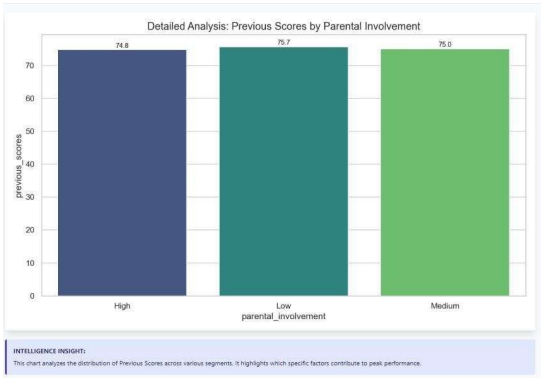

Fig. 9 Automated correlation visualization generated by the Visual Discovery Agent

 ©IJSRED: All Rights are Reserved

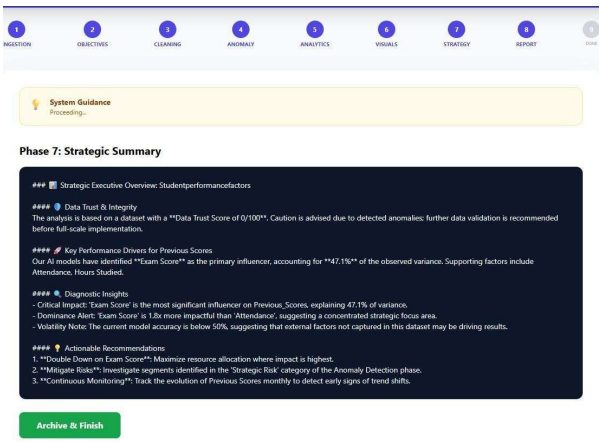

Fig. 10 Strategic narrative report with executive summary and SWOT analysis

Y. E. Discussion

Experimental results confirm that the multi-agent architecture successfully automates tasks traditionally performed by human data analysts. Reducing report generation time from hours to under ten seconds represents a substantial operational efficiency gain. Deterministic preprocessing ensures that LLM outputs remain grounded in verified statistics rather than speculative inference.

The decentralized agent design outperformed preliminary single-model prototypes in both data integrity and debuggability. Isolated agent failures could be localized and corrected without disrupting the entire pipeline. The human-in-the-loop model preserves user oversight by presenting intermediate aggregates before final narrative synthesis.

Several limitations warrant acknowledgment. The current implementation supports only structured CSV and XLSX formats; unstructured sources such as PDF reports or email archives remain unsupported. Performance partially depends on Gemini API latency and requires stable internet connectivity. Datasets with extremely high column counts demand more aggressive aggregation to remain within LLM token limits.

Compared with incumbent self-service BI tools, Agentic BI trades manual dashboard flexibility for autonomous end-to-end delivery. Organizations requiring highly customized visual layouts may still prefer traditional platforms. However, for recurring operational reporting where speed and narrative interpretation matter more than pixel-level chart customization, the agentic approach offers compelling advantages.

VII. VII. CONCLUSION AND FUTURE SCOPE

This paper presented Agentic BI, an autonomous multi-agent framework for end-to-end business intelligence automation. By orchestrating seven specialized agents over a shared state controller with Google Gemini reasoning integration, the system transforms raw structured datasets into prescriptive executive reports within sub-10-second latency on enterprise-scale benchmarks.

Key contributions include a decoupled three-tier architecture, deterministic statistical pipelines coupled with constrained LLM synthesis, autonomous data governance through self-healing imputation and outlier detection, and empirical validation demonstrating 100 percent test pass rate across thirteen enterprise scenarios. The framework democratizes advanced analytics for non-technical stakeholders who previously depended on specialized data science resources.

Z. A. Future Research Directions

Future work will extend the platform along four primary axes. First, natural language query and voice-based BI interfaces leveraging Gemini multimodal capabilities will enable executives to interrogate datasets conversationally. Second, real-time database and cloud warehouse connectors will replace static file uploads with continuous streaming anomaly alerts via email or Slack integration.

Third, prescriptive simulation agents will support what-if scenario modeling for pricing, inventory, and market strategy decisions. Fourth, multi-source data fusion will combine structured transactional records with unstructured sentiment data from social media and customer feedback channels. Role-based collaborative workspaces with access control will support enterprise deployment across departmental boundaries.

As agentic AI matures, autonomous analytics platforms such as Agentic BI may shift the role of human analysts from routine report generation toward strategic model governance and domain-

 ©IJSRED: All Rights are Reserved

specific policy definition—marking a fundamental transition in how organizations convert data into actionable intelligence.

ACKNOWLEDGMENT

The authors thank Dr. P. Chiranjeevi, Professor and Head, Department of Computer Science and Engineering, Amrita Sai Institute of Science and Technology, for guidance and support throughout this research.

REFERENCES

[1] [1] S. Few, Information Dashboard Design: Displaying Data for At-a-Glance Monitoring, 2nd ed. Burlingame, CA, USA: Analytics Press, 2019.
[2] [2] B. Arora, "Generative AI in business intelligence: Transforming data into narrative," J. Artif. Intell. Res., vol. 45, no. 2, pp. 112-128, 2023.
[3] [3] W. McKinney, "Data structures for statistical computing in Python," in Proc. 9th Python Sci. Conf., 2010, pp. 51-56.
[4] [4] A. Ramirez, "The evolution of agentic AI: From chatbots to autonomous business analysts," in Proc. Int. Conf. Mach. Learn. Appl. (ICMLA), 2025, pp. 88-95.
[5] [5] Google DeepMind, "Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context," Google DeepMind, Tech. Rep., 2024.
[6] [6] M. Chen and J. Lin, "Multi-agent systems for data science: A survey of collaborative AI," ACM Comput. Surv., vol. 54, no. 3, pp. 1-35, 2021.
[7] [7] R. Kimball and M. Ross, The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling, 3rd ed. Indianapolis, IN, USA: Wiley, 2013.
[8] [8] S. Tiwar, FastAPI: Modern Python Web Development. Sebastopol, CA, USA: O'Reilly Media, 2024.
[9] [9] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," J. Mach. Learn. Res., vol. 12, pp. 2825-2830, 2011.
[10] [10] A. Vaswani et al., "Attention is all you need," in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2017, pp. 5998-6008.
[11] [11] T. Brown et al., "Language models are few-shot learners," in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2020, pp. 1877-1901.
[12] [12] J. Nielsen, "Usability inspection methods," in Proc. CHI Conf. Hum. Factors Comput. Syst., 1994, pp. 413-414.
[13] [13] J. D. Hunter, "Matplotlib: A 2D graphics environment," Comput. Sci. Eng., vol. 9, no. 3, pp. 90-95, 2007.
[14] [14] H. Wickham, "Tidy data," J. Stat. Softw., vol. 59, no. 10, pp. 1-23, 2014.
[15] [15] M. Zaharia et al., "Apache Spark: A unified engine for big data processing," Commun. ACM, vol. 59, no. 11, pp. 56-65, 2016.

 ©IJSRED: All Rights are Reserved